Alonzo Church, jego rachunek i teza

Antoni Kościelski

Profesor Alonzo Church urodził się 14 czerwca 1903 roku w Waszyngtonie. Pochodził z rodziny, w której zwykle jeden z synów miał na imię Alonzo: pradziadek, Alonzo Church, był profesorem matematyki i astronomii w college'u, który później przekształcił się w University of Goergia; dziadek, Alonzo Webster Church, pracował między innymi w bibliotece Senatu Stanów Zjednoczonych; ojcem był Samuel Robbins Church, który – dopóki pozwalało mu zdrowie – był sędzią miejskim Dystryktu Columbia, ale pomocy w kształceniu dzieci udzielał mu dobrze sytuowany brat, Alonzo Church. Alonzo Church, bohater tego artykułu, w 1924 roku ożenił się z pielęgniarką o polsko brzmiącym nazwisku Mary Julia Kuczinski. Byli nierozłącznym małżeństwem przez 51 lat, a w 1929 roku urodził się im syn, który również został nazwany Alonzo. Zawiłości genealogiczne rodziny Churchów nie będą nas więcej zajmowały i dalej Alonzo Church będzie osobą, o której piszę w pierwszym zdaniu tego artykułu.

Wspomniany zwyczaj być może byłby wystarczającym powodem przedstawienia profesora w innych czasopismach. W *Matematyce* jednak zwykle są prezentowane sylwetki matematyków. Jest więc istotne, czy Alonzo Church był matematykiem. Na to pytanie też nie ma prostej odpowiedzi. W jego przypadku trudno przytoczyć twierdzenie świadczące o osiągnięciach niewątpliwie matematycznych. Zajmował się zagadnieniami z pogranicza matematyki. Uważam, że był jednak matematykiem, a z całą pewnością miał wykształcenie matematyczne.

Studia rozpoczął na Uniwersytecie w Princeton w wieku 17 lat. W ich trakcie napisał pierwszą pracę naukową *Uniqueness of the Lorentz Transformation*. W 1924 roku uzyskał stopień Bachelors of Art w zakresie matematyki. Nie rozstał się z uczelnią i kontynuował studia pod kierunkiem Oswalda Veblena[1], bardzo wpływowej postaci w środowisku matematycznym w Princeton. W 1927 roku przedstawił dysertację *Alternatives to Zermelo's Assumption*, w której nawiązywał do badań teoriomnogościowych swego promotora, i uzyskał stopień doktora.

Założenie Zermelo to nic innego jak aksjomat wyboru. W swojej dysertacji Church analizuje konsekwencje trzech postulatów, w tym sprzecznych z aksjomatem wyboru. Jest tam stwierdzenie, że nie jest nieprawdopodobne, by żaden z tych postulatów nie prowadził do sprzeczności, i może być wiele teorii mnogości podobnie, jak jest możliwe wiele geometrii zależnie od tego, czy przyjmiemy bądź odrzucimy piąty postulat Euklidesa. Ciekawe jest też pierwsze zdanie dysertacji stwierdzające, że *celem tej pracy jest rozważenie możliwości stworzenia logiki, w której aksjomat wyboru nie zachodzi*. Słowo logika miało wtedy szersze znaczenie niż obecnie i obejmowało teorię mnogości .

[1] Oswald Veblen był prezydentem American Mathematical Society w latach 1923 - 24. Brał udział w tworzeniu Szkoły Matematycznej w słynnym Institute of Advanced Study i był pierwszą osobą tam zatrudnioną.

Początki rachunku lambda. Doktorat został uhonorowany dwuletnim stypendium National Research Fellowship. Dzięki temu, po krótkim okresie pracy na uniwersytecie w Chicago, Church odwiedził Harward, a następnie udał się do Europy. W Getyndze współpracował z matematykami skupionymi wokół Dawida Hilberta, a w Amsterdamie pracował razem z Luitzenem Brouwerem i Arendem Heytingiem. Dzięki podróży osobiście poznał wielu europejskich logików. Po powrocie związał się na wiele lat z Uniwersytetem w Princeton. Swoje rozważania z tego okresu przedstawił w pracy *A Set of Postulates for the Foundation of Logic* z 1932 roku. Jest to historyczna praca, w której mimochodem zostały sformułowane zasady rachunku lambda.

We wstępie do niej znajdujemy mało zrozumiałą opinię, która często pojawiała się w tamtych czasach: *Przedstawimy zbiór postulatów dla podstaw logiki formalnej, w których unikamy używania zmiennych wolnych i w których – by uniknąć paradoksów związanych z matematyką pozaskończoną – wprowadzamy pewne ograniczenia na prawo wyłączonego środka. Nie chcemy używać zmiennych wolnych, gdyż wymagamy, by każda kombinacja symboli należących do naszego systemu, jeżeli w ogóle reprezentuje zdanie, reprezentowała określone zdanie, nie wymagające dodawania dodatkowych ustnych wyjaśnień. Mamy nadzieję, że jest widoczne, iż używanie zmiennych wolnych narusza to wymaganie. Na przykład równość*

(1) $a(b+c)=ab+ac$

w której a, b i c są używane jako zmienne wolne, nie przedstawia określonego zdania poza przypadkiem, gdy jest wiadome, jakie wartości mogą być wzięte za te zmienne, i te informacje, jeżeli nie wynikają z kontekstu, muszą być słownym dodatkiem. Dopuszcza się, by zakres zmiennych a, b i c mógł składać się z wszystkich liczb rzeczywistych, lub z wszystkich liczb zespolonych, lub z elementów jakiegoś innego zbioru, lub też że zakresy tych zmiennych mogą sie różnić. W każdej z tych możliwości równanie (1) ma inne znaczenie. Jest jasne, że gdy samo to równanie zostało napisane, zamierzone zdanie nie zostało całkowicie przetłumaczone na język symboliczny i w celu całkowitego tłumaczenia, konieczne słowne dodatki muszą być wyrażone za pomocą symboli logiki formalnej i zawarte razem z równaniem w formule użytej do reprezentowania zdania.

Bardziej interesujący niż sam zbiór postulatów okazał się zastosowany sposób formalizacji logiki i unikania zmiennych wolnych, czyli rachunku lambda. Jest to rachunek formalny: piszemy w nim specyficzne wzory i przekształcamy je zgodnie z ustalonymi zasadami. Zwykle jest przedstawiany przez odwołanie się do intuicji związanych z funkcjami, ale nie zawsze jest z nimi zgodny. Aby zapoznać się choć trochę z tym rachunkiem, musimy inaczej myśleć o funkcjach i poznać specyficzne zasady notacyjne.

Formalizm rachunku lambda. Bardzo często mamy do czynienia z funkcją f i jej argumentem x. W takich sytuacjach interesuje nas zwykle wartość funkcji dla wskazanego argumentu, oznaczana najczęściej przez $f(x)$. Możemy jednak pomyśleć, że stosujemy dwuargumentową operację, która funkcji f i argumentowi x przypisuje jako wynik $f(x)$. Tę operację nazywamy aplikacją i sporadycznie oznaczamy kropką, podobnie jak mnożenie. Możemy więc napisać

$f \cdot x = f\,x = f(x)$,

co znaczy, że aplikując funkcję f do argumentu x otrzymujemy jako wynik wartość $f(x)$, a ponadto, jeżeli to tylko jest możliwe, opuszczamy kropkę oznaczającą działanie,[2] podobnie jak to czynimy z kropką oznaczającą mnożenie.

Przytoczona definicja aplikacji dotyczy funkcji jednoargumentowych, ale posługujemy się też funkcjami z wieloma argumentami. Wtedy stosujemy ogólniejsze pojęcie aplikacji, które rozumiemy jako ustalenie wartości pierwszego argumentu. Dla funkcji dwóch zmiennych f aplikacja $f\ x$ oznacza funkcję g zdefiniowaną wzorem $g(y) = f(x, y)$. Na przykład napis $+\ 3$ oznacza funkcję, która argumentowi y przyporządkowuje liczbę $3 + y$, a więc operację dodawania do 3, a dopiero gdy tę operację zaaplikujemy do liczby 1, otrzymujemy sumę 3 i 1, czyli 4.

W powyższym przykładzie aplikację stosujemy dwukrotnie. Wtedy musimy pamiętać, że nie jest ona łączna i zwykle wyrażenia

$(f \cdot x) \cdot y$ oraz $f \cdot (x \cdot y)$

mają różne wartości. Nie możemy więc swobodnie opuszczać nawiasów tak, jak to robimy w przypadku analogicznych wyrażeń z mnożeniem. Z drugiej strony, nadmiar nawiasów ogranicza czytelność wzorów. Przyjmuje się więc, że mamy prawo opuścić nawiasy w pierwszym z podanych wyrażeń. Możemy je skrócić nawet do postaci fxy. W ogólniejszym przypadku, napis $fxyz$ jest skróconą wersją wyrażeń $((f \cdot x) \cdot y) \cdot z$ oraz $((f\,x)\,y)\,z$.

Możemy też napisać $+\ 3\ 1\ 2$, czyli $((+\ 3)\ 1)\ 2$. Wartość tego wyrażenia powinniśmy otrzymać aplikując 4 do liczby 2. Nie wiadomo jednak, co w ten sposób otrzymamy. Być może powinniśmy uznać to wyrażenie za niepoprawne. A może powinniśmy najpierw ustalić, czym jest 4, a jeżeli okaże się jakąś funkcją, to także może da się ją zaaplikować do liczby 2. Niejasne jest również znaczenie wzoru $+\ (3\ 1)$, w którym dodawanie jest aplikowane do czegoś otrzymanego w wyniku aplikowania 3 do 1.

Tak naprawdę w rachunku lambda nie ma liczb naturalnych, choć mogą być w nim reprezentowane. Analizujemy w nim abstrakcyjną operację aplikacji i – mając dowolne x i y – możemy aplikować x do y, a także y do x.

Eliminacja zmiennych. Uczonym, który w 1920 roku zaproponował pewien sposób eliminowania zmiennych wolnych, jest Mojżesz Schönfinkel, Rosjanin wykształcony w Odessie, który przez dziesięć lat był w zespole Dawida Hilberta w Getyndze. Jego nazwisko pojawiło się na dwóch pracach. W pierwszej do eliminowania zmiennych użył tak zwanych kombinatorów. Ważnymi przykładami są kombinatory S i K, o których przyjmujemy, że aplikowane mają własności wyrażone wzorami

$S\,x\,y\,z = x\,z\,(y\,z)$ oraz $K\,x\,y = x$.

Dla przykładu przyjrzyjmy się prawu wyłączonego środka. To prawo logiczne wyraża związek alternatywy i negacji. Zwykle zapisujemy je w postaci

$p \vee \neg p$,

[2] Taka zasada jest bardzo wygodna, ale sprawia kłopoty, gdy chcemy korzystać z wieloznakowych nazw takich, jak *sin*. W tym artykule nie będziemy tego robić, a przykładowe liczby będą jednocyfrowe.

używając zmiennej wolnej, w tym przypadku *p*. Tworzymy je stosując (aplikując) alternatywę do zmiennej *p* i formuły otrzymanej przez zanegowanie *p*. Możemy je zapisać, zależnie od przyjętej konwencji (logicznej, funkcyjnej lub aplikacyjnej), w najróżniejszych wersjach

$p \vee \neg p \; = \; \vee(p, \neg(p)) \; = \; \vee\, p\, (\neg\, p)$.

Nietrudno zauważyć, że ostatnia, aplikacyjna postać prawa wyłączonego środka może też zostać formalnie zapisana z użyciem kombinatora *S* jako

$S \vee \neg\, p$.

Schönfinkel proponował, by prawem wyłączonego środka nazywać wyrażenie $S \vee \neg$ bez zmiennej wolnej, przedstawiające pewien związek między alternatywą v i negacją ¬ wyrażony za pomocą kombinatora *S*. Dopiero, gdy to prawo zostanie zaaplikowane do konkretnego zdania *p*, otrzymamy jego szczególny egzemplarz postaci $\vee\, p\, (\neg\, p)$.

Pomysł Schönfinkela można analizować badając algebry z dwuargumentowym działaniem, które nazywamy aplikacją, i w których są elementy *S* i *K* spełniające wyżej podane prawa. Takie algebry istnieją, także nie jednoelementowe, ale podanie istotnego przykładu nie jest rzeczą banalną. Gdyby przyjąć, że wszystkie spójniki logiczne i zmienne zdaniowe są elementami takiej algebry, to każdą formę zdaniową dałoby się w niej przedstawić podobnie, jak prawo wyłączonego środka: jako wyrażenie zbudowane z kombinatorów *S* i *K* zaaplikowane do spójników i zmiennych występujących w formie. Mamy na przykład

$S\,(S(KS)K)\,(SKK)\,f\,x \; = \; f\,(f\,x)$,

a gdyby za *f* w tym wzorze wziąć operację negacji, to otrzymalibyśmy formę, która jest podwójnym zaprzeczeniem zmiennej *x*.

Rachunek lambda. Wróćmy do naszego bohatera. Alonzo Church uzupełnił[3] pomysł Schönfinkela o operację abstrakcji. W monografii *The calculi of Lambda-Conversion* tak wprowadza tę operację: *Aby wziąć przykład z teorii funkcji liczb naturalnych, rozważmy wyrażenie $(x^2+x)^2$. Jeżeli mówimy „$(x^2+x)^2$ jest większe niż 1000”, to wypowiadamy zdanie, które zależy od x i nic nie znaczy, chyba że x jest znane jako szczególna liczba naturalna. Z drugiej strony, jeżeli mówimy „$(x^2+x)^2$ jest funkcją pierwotnie rekurencyjną”, to wypowiadamy dobrze określone zdanie, które w żaden sposób nie zależy od znaczenia zmiennej x. Różnica między tymi przypadkami jest taka, że w pierwszym wyrażenie $(x^2+x)^2$ służy za wieloznaczne lub zmienne określenie liczby naturalnej, podczas gdy w drugim jest określeniem szczególnej funkcji. Będziemy je odtąd rozróżniać używając $(x^2+x)^2$, gdy myślimy o wieloznacznej liczbie naturalnej, oraz $\lambda x\,(x^2+x)^2$ jako oznaczenie odpowiedniej funkcji. ... Symbol λx nazywamy operatorem abstrakcji i mówimy o funkcji oznaczanej przez λx M, że została otrzymana z wyrażenia M przez abstrakcję. ... Wyrażenie λx (λy M), które będziemy często skracać do λxy. M, oznacza funkcję, której wartość dla argumentu x jest dana wzorem λy M, albo też funkcję dwóch zmiennych.*

W rachunku lambda, oczywiście, możemy również aplikować funkcje definiowane przez abstrakcję. Zgodnie z intuicyjnym

[3] Może to stwierdzenie jest logicznie uzasadnione, ale niekoniecznie odpowiada prawdzie historycznej.

określeniem abstrakcji i podpowiedziami Churcha można spodziewać się, że aplikując funkcję $\lambda x\ (x^2+x)^2$ do 3 otrzymujemy wartość wyrażenia definiującego funkcję $(x^2+x)^2$, w którym zamiast x podstawiamy 3, a więc $(\lambda x\ (x^2+x)^2)\ 3\ =\ (3^2+3)^2$.

Jeżeli aplikujemy abstrakcję $\lambda x\ M$ do N, to otrzymujemy wyrażenie powstające z M poprzez zastąpienie zmiennej x wyrażeniem N. Aplikując tak zdefiniowane funkcje wykonujemy więc przekształcenie zwane podstawianiem[4], bardzo często używane przez matematyków. Przekształcenie wyrażenia $(\lambda x\ M)\ N$ we wskazane podstawienie nazywamy β-redukcją.

Wyrażenia rachunku lambda tworzymy ze zmiennych używając dwóch operacji: abstrakcji i aplikacji. Takie wyrażenia przekształcamy stosując β-redukcję. Otrzymujemy w ten sposób rachunek formalny, w którym możemy dowodzić równości wyrażeń. W szczególności, równe są wyrażenia $(\lambda x\ M)\ N$ i odpowiednie podstawienie otrzymane w wyniku β-redukcji. Dalsze równości wynikają z ogólnych własności tej relacji, przyjmowanych w całej matematyce, i z otrzymanych przez β-redukcję.

Trudno wyobrazić sobie, czemu ten rachunek miałby służyć. Początkowo miał stanowić podstawę sformalizowanej logiki. Widać jednak, że możemy w nim przeprowadzać jakieś obliczenia, na przykład dla $S = \lambda xyz.\ xz(yz)$ oraz $K = \lambda xy.\ x$ wyrażenie $S\ (S(KS)K)\ (SKK)$ daje się przekształcić – z dokładnością do nazw zmiennych – do postaci $\lambda zt.\ z(zt)$. Ma też własności odbiegające od intuicji. W szczególności, dla $I = \lambda x.\ x$ w rachunku lambda zachodzi równość $I\ I = I$, podczas gdy w matematyce opartej na teorii mnogości żadnej funkcji nie możemy aplikować do samej siebie.

Odkrywanie tezy Churcha. Alonzo Church zdawał sobie sprawę, że w pracy o postulatach logiki proponuje system, o którym nie potrafi pokazać, że jest niesprzeczny i pozwala na unikanie paradoksów. Obiecuje więc tylko, że w razie potrzeby zostanie poprawiony. W krótkim czasie odkrywa sprzeczność w swoim systemie, a nowy zbiór postulatów przedstawia w kolejnej pracy z 1933 roku, o takim samym tytule. Ma jednak pecha. Po pewnym czasie jego uczniowie Stephen Kleene i Barkley Rosser wyprowadzają sprzeczność także ze zmienionego zbioru postulatów.

Ale ta druga praca Churcha też ma w sobie coś interesującego. Są w niej zawarte wyrażenia rachunku lambda, które mogą reprezentować liczby naturalne, tak zwane numerały Churcha, definiowane wzorami

$C_0 = \lambda fx.\ x,\quad C_1 = \lambda f\,x.\ f\,x,\quad C_2 = \lambda fx.\ f\ (f\ x),\ \ldots$

Ogólnie, $C_n = \lambda f\,x.\ f\ (f\ (\ldots\ (f\ x)\ldots)))$ jest dane wzorem, w którym symbol f jest n-krotnie aplikowany. Są tam też podane równości

[4] Podstawianie wykonujemy na co dzień, ale nie jest to operacja banalna. Najpierw zauważmy, że aplikując do czegokolwiek funkcje $\lambda x\ f(f\ x)$ i $\lambda y\ f(f\ y)$ otrzymujemy to samo wyrażenie. Nie ma więc powodu rozróżniania tych funkcji. Zmienną x w wyrażeniu $\lambda x\ f\ (f\ x)$ nazywamy związaną operatorem λ, a zmienną f – wolną (niezwiązaną żadnym operatorem). W wyrażeniu $(\lambda fx.\ f\ (f\ x))\ x$ zmienna x jest zarówno wolna, jak i związana (to x w nawiasach). Przekształcenia w rachunku lambda muszą być tak prowadzone, aby zmienne wolne nie stawały się związanymi w wyniku podstawiania, podczas takich przekształceń, jak $(\lambda fx.\ f\ (f\ x))\ x = (\lambda f\ (\lambda x\ f\ (f\ x)))\ x = \lambda x\ x\ (x\ x)$. W tej sytuacji powinniśmy najpierw zmienić nazwę zmiennej związanej x, a później wykonać redukcję: $(\lambda fx.\ f(f\ x))\ x = (\lambda f\ (\lambda x\ f\ (f\ x)))\ x = (\lambda f\ (\lambda y\ f\ (f\ y)))\ x = \lambda y\ x\ (x\ y)$. Podobne kłopoty mamy w rachunku kwantyfikatorów, a także w rachunku całkowym.

$(\lambda xypq.\, xp(ypq))\ C_m\ C_n = C_{m+n}$ oraz $(\lambda xyz.\, x(yz))\ C_m\ C_n = C_{m\cdot n}$.

Okazało się bowiem, że w rachunku lambda można przeprowadzać obliczenia na liczbach naturalnych, w szczególności można dodawać je, mnożyć, także potęgować. Biorąc na przykład $\lambda xyz.\, x(yz)$ i dopisując do tego wyrażenia C_m i C_n reprezentujące liczby *m* i *n*, a następnie wszystko to przekształcając, możemy otrzymać wyrażenie $C_{m\cdot n}$ przedstawiające iloczyn $m\cdot n$.

Przyjmijmy taką oto definicję: funkcja naturalna *f* (ustalmy, że dwóch zmiennych) jest definiowalna w rachunku lambda, jeżeli dla pewnego wyrażenia *F* tego rachunku i dla wszystkich liczb naturalnych *m* i *n* zachodzą równości $F\ C_m\ C_n = C_{f(m,\, n)}$. W ten sposób dodawanie i mnożenie liczb naturalnych uznajemy za funkcje definiowalne w rachunku lambda.

Dodawać i mnożyć potrafimy bez rachunku lambda, a sam ten rachunek raczej takie obliczenia komplikuje. Mimo to dostarcza formalną metodę prowadzenia obliczeń, stosunkowo łatwą do wykorzystania w rachunkach komputerowych. Ciekawe jest więc pytanie, jakie funkcje są definiowalne w rachunku lambda i – w związku z tym – mogą być automatycznie obliczane. Alonzo Church odpowiedział na to pytanie, choć nieco inaczej umotywowane, formułując słynną tezę Churcha. Teza ta charakteryzuje klasę funkcji efektywnie obliczalnych. Jest to bardzo ważna klasa funkcji naturalnych, tych, które są obliczalne w sensie intuicyjnym, za pomocą jakkolwiek rozumianych algorytmów, mechanicznych i skończonych procedur, a także wszelkich innych dających się pomyśleć efektywnych sposobów. Klasa ta ma mało precyzyjne określenie, ale musiał ją dokładnie opisać każdy matematyk, który – podobnie jak Church – chciał negatywnie rozwiązać tzw. Entscheidungsproblem postawiony przez Dawida Hilberta.

Łatwo zauważyć, że funkcje definiowalne w rachunku lambda powinny być efektywnie obliczalne. Potrafimy przecież prowadzić obliczenia w tym rachunku. Church przeczuwał, że zachodzi także zawieranie odwrotne, i efektywnie obliczalne funkcje są definiowalne w rachunku lambda. Ale był też fakt świadczący o czymś przeciwnym. Przez długi czas nikt nie potrafił dowieść definiowalności poprzednika, funkcji przyporządkowującej dodatniej liczbie naturalnej wartość o jeden mniejszą. Dowód w końcu znalazł Stephen Kleene w 1932 roku. Anegdota głosi, że miało to miejsce podczas zabiegu dentystycznego przeprowadzanego pod wpływem podtlenku azotu lub, jak kto woli, gazu rozweselającego, substancji znieczulającej o działaniu narkotycznym, i był to pierwszy rezultat naukowy młodego studenta. Wtedy – być może – przeczucia Churcha zaczęły przeradzać się w jasno sformułowaną hipotezę naukową. Po latach Stephen Kleene miał opowiadać trochę z żalem, ale i z podziwem dla swego mistrza: *Chciałbym móc powiedzieć, że w momencie odkrycia, jak lambda definiować funkcję poprzednika, miałem ideę tezy Churcha. Ale nie miałem, a Church ją miał.*

W 1931 roku ukazała się i została bez zwłoki przedstawiona w Princenton przez Johna von Neumanna praca Kurta Gödla o niezupełności arytmetyki. Dostarczyła kolejnego impulsu w badaniach Churcha. Doceniając znaczenie pracy, Oswald Veblen zaprosił jej autora do Institute of Advanced Study. Na początku 1934 roku, od lutego do maja, Kurt Gödel osobiście przedstawiał swoje rezultaty. Treść wykła-

du została starannie zanotowana przez studentów Stephena Kleene'ego i Barkleya Rossera i nawet dzisiaj możemy się z nim zapoznać. Znajduje się w nim stwierdzenie, że funkcje pierwotnie rekurencyjne, użyte w dowodzie twierdzenia o niezupełności, mogą być obliczane za pomocą skończonych procedur, a po odpowiednim rozszerzeniu definicji – powinno być też prawdziwe stwierdzenie odwrotne. Kurt Gödel sformułował więc hipotezę przypominającą przemyślenia Churcha. Uważał ją jednak za mało dokładną i słabszą od tezy Churcha. W rozmowie z Churchem odniósł się sceptycznie do możliwości opisania wszystkich funkcji efektywnie obliczalnych. Obiecał jednak, że zaproponuje obszerniejszą definicją rekurencyjności i dotrzymał słowa. Kończąc swój wykład, naszkicował definicję klasy funkcji, które dzisiaj nazywa się rekurencyjnymi według Herbranda i Gödla[5].

Church wykorzystał to niezwłocznie. Pracując ze Stephenem Kleene'em i Barkleyiem Rosserem ustalił podstawowe fakty o rachunku lambda i funkcjach rekurencyjnych. Razem z Rosserem dowiódł twierdzenie Churcha – Rossera o postaci normalnej. Wynika z niego, że w rachunku lambda nie można wyprowadzić równości numerałów Churcha reprezentujących różne liczby naturalne, a więc że nie jest to rachunek, który tylko mówi coś o pojedynczym obiekcie, i zachodzi w nim każda równość.

[5] Intencją Kurta Gödla było zdefiniowanie klasy funkcji zamkniętej ze względu na wszelkie definicje rekurencyjne (sposoby definiowania przez indukcję). Dzisiaj funkcje rekurencyjne najczęściej definiujemy używając tzw. operacji minimum (najmniejsze n takie, że …). Współczesna definicja, Kleene'ego, jest łatwiejsza w użyciu, ale przestał być widoczny jej związek z rekursją. Oczywiście, obie definicję są równoważne.

Zostało też pokazane, w znacznym stopniu przez Kleene'ego, że funkcje rekurencyjne według Herbranda i Gödla to dokładnie funkcje definiowalne w rachunku lambda. W ten sposób Church otrzymał ważny argument przemawiający za jego hipotezą: dwie odmiennie umotywowane próby formalizacji obliczalności okazały się równoważne.

Wyniki badań Church ogłosił na posiedzeniu American Mathematical Society w dniu 19 kwietniu 1935 roku. Zaproponował wtedy utożsamienie efektywnej obliczalności z pojęciem funkcji rekurencyjnej i przedstawił przemawiające za tym argumenty. Krótko po tym zdanie stwierdzające równoważności tych dwóch pojęć zaczęło być nazywane tezą Churcha. Tezą, ponieważ nie można go dowieść, choć można przedstawić wiele argumentów świadczących o jego słuszności.

19 kwietnia 1935 roku to moment historyczny. Narodziła się wtedy informatyka teoretyczna, a matematycy zyskali możliwość dowodzenia twierdzeń mówiących, że niektórych zadań nie można rozwiązywać w sposób zalgorytmizowany. Wśród tych zadań jest

Entscheidungsproblem, który miał *zostać rozwiązany, gdy będzie znana procedura, za pomocą której będzie można decydować w skończonej liczbie operacji, czy dane wyrażenie logiczne jest ogólnie prawdziwe, bądź czy jest spełnialne*.

Dawid Hilbert był przekonany o możliwości opracowania takiej procedury. Jednak już pierwsze przykłady funkcji nierekurencyjnych podane przez Churcha świadczyły dobitnie, że takiej procedury nie ma. Church bez trudu przeformułował twierdzenie o niezupełności otrzymując twier-

dzenie o nierozstrzygalności arytmetyki. Pokazał więc, że nie ma zalgorytmizowanej procedury pozwalającej ustalić, czy dane zdanie arytmetyczne daje się wywnioskować z aksjomatów Peano. Następnie wzmocnił ten rezultat dowodząc analogiczne twierdzenie o nierozstrzygalności rachunku kwantyfikatorów i tym samym rozwiązał Entscheidungsproblem.

W 1936 roku Church opublikował jeszcze razem z Stephenem Kleene'm w polskich *Fundamenta Mathematicae* pracę poświęconą najmniejszej nierekurencyjnej liczbie porządkowej. Nieco później stworzył rachunek lambda z typami, odgrywający dzisiaj – razem z czystym rachunkiem lambda –istotną rolę w teorii języków programowania. Powoli jednak przestawał zajmować się zagadnieniami związanymi z matematyką i teoretyczną informatyką, a jego zainteresowania zaczęły przesuwać się w stronę logiki i filozofii. Wśród opowieści o nim jest też i taka z początku lat osiemdziesiątych: podczas wizyty w Stanford zostały mu zaprezentowane komputery wykorzystujące LISP, język funkcyjny oparty o rachunek lambda. Church nie wykazał zainteresowania, a tłumacząc się z ewidentnego braku entuzjazmu powiedział, że nie zna się na komputerach ani trochę, ale miał kiedyś studenta, który wie o nich co nieco. Mówiąc to miał na myśli Alana Turinga.

Turing i inni uczniowie. Turing był rzeczywiście uczniem Churcha, ale raczej nie studentem. Otrzymał z rąk Churcha tytuł doktora. Był to uczeń specyficzny, dorównujący mistrzowi, może nawet przewyższający go, choć wyraźnie młodszy. Jako student uzyskał analogiczne rezultaty i ujawnił je tuż przed ukazaniem się pracy z tezą Churcha. Dzięki temu znalazł się w Princeton i tam przygotował rozprawę doktorską.

W sumie Alonzo Church wypromował 31 doktorów. Wśród nich jest wielu wybitnych uczonych, także odgrywających znaczącą rolę w rozwoju informatyki i jej teoretycznych podstaw. Są to między innymi wspomniani Stephen Kleene i Barkley Rosser, a także znani już Czytelnikom *Matematyki* Martin Davis, Dana Scott i Michael Rabin (*Matematyka*, 4/2009 i 9/2007).

Działalność wydawnicza. Church był nie tylko uczonym. Był też jednym z założycieli *Association of Symbolic Logic*, towarzystwa naukowego skupiającego logików, utworzonego w 1936 roku, i *Journal of Symbolic Logic*, czasopisma wydawanego przez to stowarzyszenie. Był pierwszym redaktorem tego czasopisma, ukształtował je i redagował przez kilkadziesiąt lat. Rzeczą charakterystyczną jest to, że *Journal of Symbolic Logic* dostarczał informacji o wszelkich pracach z dziedziny logiki. W pierwszym tomie można znaleźć *Bibliography of symbolic logic*, zebrany przez Churcha wykaz wszelkich znanych mu nowożytnych prac z logiki napisanych przed 1936 rokiem. Zaczynał się pracą Gottfrieda Leibniza z 1666 roku, był uzupełniany przez wiele lat, a dzisiaj stanowi ważne źródło wiedzy o rozwoju logiki.

Dziełem Churcha jest też *Rewievs Section* w *Journal of Symbolic Logic*. W tym dziale były zamieszczane krótkie informacje, recenzje, omówienia, także proste dane bibliograficzne wszelkich – w zamyśle – artykułów i monografii z logiki symbolicznej. Niewielka redakcja tego działu przeglądała nawet około 300 czasopism z całego świata w poszukiwaniu interesujących

prac logicznych, śledziła ukazujące się monografie i wynajdowała recenzentów.

Alonzo Church redagował prace publikowane w *Journal of Symbolic Logic* w latach 1936 – 1950, czyli przez lat 15. Za *Reviews Section* odpowiadał przez 44 lata, do roku 1979. Ostatniego doktora wypromował w roku 1986, kształcił więc przyszłe pokolenia badaczy przez pół wieku. W 1929 roku rozpoczął pracę na Uniwersytecie w Princeton, w 1947 został pełnym profesorem, w 1967 przeniósł się do Los Angeles i objął stanowiska *Flint Professor of Philosophy* i *Professor of Mathematics*. Uczył do roku 1990 i przeszedł na emeryturę w wieku 87 lat, po 60 latach pracy. Ostatnią pracę opublikował w 1995 roku, pracował twórczo przez 70 lat. Zmarł 11 sierpnia 1995, w wieku 92 lat. Po niezwykle pracowitym życiu spoczął na Princeton Cementery obok żony i rodziców, a także Kurta Gödla i Johna von Neumanna.

Herbert B. Enderton (1998): Alonzo Church and the Rewievs, Biulletin of Symbolic Logic, 4:2, 172-180

Herbert B. Enderton: Alonzo Church: his life and work

Maía Manzano (1997): Alonzo Church: his life, his work and some of his miracles, History and
Philosophy of Logic, 18:4, 211-232

What happened one day during the academic year of 1983-84 was as follows. Alonzo Church had been to give a talk at the CSLI. At the time, CSLI had a large number of Xerox Dandelion computers, which were LISP processing machines (similar to Symbolics machines, which became popular a few years later). Etchemendy took Church aside for a few minutes to show him a Dandelion and to explain that it was indirectly based on lambda calculus. He seemed only distantly interested and when they later walked to the building in which Church was to give his talk, he told Etchemendy (no doubt to explain his evident lack of enthusiasm about the demo) that he knew nothing at all about computers, but that he had once had a student who did know quite a bit about them. That student was Alan Turing.

CSLI - Center for the Study of Language and Information

The Entscheidungsproblem is solved when one knows a procedure by which one can decide in a finite number of operations whether a given logical expression is generally valid or is satisfiable. The solution of the Entscheidungsproblem is of fundamental importance for the theory of all fields, the theorems of which are at all capable of logical development from finitely many axioms.
D. Hilbert, W. Ackermann, Grundz¨uge der theoretischen Logik, 1928